Spring 4-2004

Autonomous Planned Path Under-Vehicle Searching System

Christopher Robert Harvey
University of Tennessee - Knoxville

Appendix E - UNIVERSITY HONORS PROGRAM
SENIOR PROJECT - APPROVAL

Name: Chris Harvey

College: Engineering Department: Electrical + Computer

Faculty Mentor: Dr. Edwards

PROJECT TITLE: Autonomous Planned Path Under-Vehicle Searching System

I have reviewed this completed senior honors thesis with this student and certify that it is a project commensurate with honors level undergraduate research in this field.

Signed: [signature] Faculty Mentor

Date: 4/30/04

General Assessment - please provide a short paragraph that highlights the most significant features of the project.

Comments (Optional):

Chris was an integral member of the path planning team for ECE 400. Path planning for autonomous navigation is a necessary evolution for next generation robotics, and Chris has contributed to solving the current problems in the field. Despite equipment failures, he still established new techniques for under vehicle inspection.

Autonomous Navigation for Under Vehicle Inspection Robot

Team Members:
Derek Agee, Chad Kiger, Chris Harvey, and Erin Gessner

Advisors:
Dr. Mongi Abidi and Dr. Laura Morris Edwards

ECE 400 Senior Design

April 28, 2004

Imaging, Robotics, and Intelligent Systems Laboratory
Dept. of Electrical and Computer Engineering
The University of Tennessee
Email: charvey1@utk.edu, co1@utk.edu, egessner@utk.edu, ckiger@utk.edu

Contents

Table of Figures

1 PROPOSAL

The main purpose of performing under-vehicle inspection is to effectively identify any threat located beneath a vehicle. However, there are many other key factors, which lead to a successful search of a vehicle. By using a robot, the efficiency and accuracy of the search can be greatly improved. Because of the high levels of technology required to produce robots of this nature, the target buyers and users will be government employees. This will include the military as well as border patrols. To improve upon the search process, an under-vehicle robot must be able to survey the entire underside of an automobile in a timely manner. Currently, operators are required to remotely control the movement of the robot while watching a video screen of the images sent back by the robot. As with any task that is humanly controlled, there is room for error. In addition, humans are not perfect in their control of the robot. As was mentioned above, efficiency is a key element in marketing a product of this level. The only way to ensure accuracy is to have a computer-controlled search system. Because of this, it is our desire to add autonomy to an under-vehicle searching robot to improve the effectiveness of the product.

2 BACKGROUND

2.1 SECURITY NEEDS

Throughout history, security has been a concern among the major powers of the world. There has always been a need to keep enemies out and allies in. This can be seen as far back as 1184 B.C. when the Greeks used the Trojan horse to conquer Troy. The only reason they were able to defeat Troy was because Troy failed to inspect the gigantic horse that was given to them as a gift. Had the Trojans known more about security, as we do now, the world might be a totally different place. While bad for the Trojans, it was a great wake up call for the rest of the world. No longer would enemies be let in through the gates so easily.

Jumping ahead to a more present day history, which is more relevant to the task at hand, security is needed in a wide variety of facilities. It is needed in most all government locations including army bases, prisons, airports, borders, nuclear plants, and other various government establishments. Although the types of locations may differ greatly, the need for security remains to only allow certain people in while keeping the unwanted out. If the wrong people were allowed access to these facilities, there could be dire consequences, some of which could lead to massive loss of life. For this reason, bombs and chemical weapons must be kept out of all secure areas. In addition to protection from physical harm, there are many other alternative needs for active security. At airports or border control centers, it is more likely that smuggling items into the country or onto a plane would be the main concern. Nonetheless, it is clear that security is an issue in a broad range of situations.

2.2 UNDER-VEHICLE SEARCHING

This project focuses primarily on the security procedures involved with how vehicles enter and/or exit a facility or border. More specifically, this project focuses on the searching of the underside of vehicles. For this reason, it is necessary to examine the historical background of under-vehicle searching. At first, security guards would merely bend over and peer underneath a vehicle to look for anything suspicious. Because of the times and overall lack of security in the nation, it was not as crucial as in today's society. While this may have been acceptable in the earlier times of horse-drawn carriages, it is not only ineffective but also completely unacceptable in the age of modernity. Unfortunately, security has only upgraded slightly over time. It was found that if a mirror was placed on the end of a pole, an observer could view a greater area of the underside without having to bend down multiple times which could help reduce back injuries of searching employees. This method is still widely used today despite its many flaws. First, many people have trouble looking into a mirror and seeing things on a small scale. Second, the pole-mirror system only allows for a viewer to survey approximately 40% of the area underneath the vehicle, excluding a majority of the center portion. To resolve some of these issues, the CSOIS (Center for Self-Organizing and Intelligent Systems) group of the Department of Electrical and Computer Engineering at Utah State University created a robot specifically made to scan the underside of a vehicle.

The robot developed was officially called ODIS (Omni-Directional Inspection System). An ODIS robot can be seen below in Figure 1. It is approximately five inches tall, three feet long and two feet wide, although the dimensions vary depending on the specific model. The ODIS in use at the University of Tennessee at Knoxville is equipped with a camera that is pointing into an angled mirror allowing it to see a greater area underneath a vehicle while keeping the height of the robot to a minimum. The camera's video feed is sent wirelessly over a network to a remote screen that allows the user to view what ODIS sees. ODIS is remotely controlled using standard R/C signals and frequencies. The robot design solved both of the major issues with searching. Operators are able to view the image clearly on a screen in addition to having the ability to search the entirety of the underneath section. It is also small enough to fit underneath even the shortest of vehicles while maintaining the ability to be moved around and continue scanning. Although ODIS has greatly improved the techniques used at security checkpoints, the major setback of ODIS remains high levels of human interaction. Humans are not perfect, therefore, the higher the level of human interaction, the greater the risk for error. Because of this, only a fully computerized system can be 100% effective.

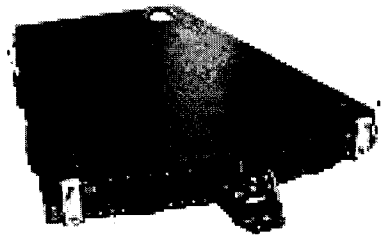


Figure 1: ODIS robot [Cso03].

3 DESIGN PROCESS

3.1 BRAINSTORMING AND IMPLEMENTATION CHOICES

The first part of the design process was choosing the direction of the research. At the beginning of the semester, there were three different routes that could be taken. The first option was to get in touch with the creators of ODIS, Utah State University, and ask for all of the necessary command codes that are implemented for robot movement. The second option was to reverse engineer all the commands by hooking the output of the controller to an oscilloscope. From there, it would be necessary to determine the output for each possible movement. Finally, the output sequences found could be used to add autonomy to ODIS. Lastly, it was an option to help research autonomy for a new robot that was in development for under-vehicle searching. This newer robot is an IRIS Laboratories creation using a tank drive system rather than omni-directional. Although the newer robot does not have an official name yet, it will be referred to as UVIS (Under-Vehicle Inspection System) to differentiate the two robots.

The first option would have been successful if Utah State freely gave its code to the public. With the code, it would have been possible to effectively program autonomy for ODIS. Due

to time constraints, the second option is impractical. Because of the nature of project, the majority, if not all of the semester, would have been spent reverse engineering the commands to control movement. At the suggestion of Dr. Page, the team chose to work with the UVIS robot. This newer version is being built with an Innovation First Microcontroller. Because of this, much of the logic, hardware, and programming can be tested on what is called an EduBot. This educational kit, created by Innovation First, comes with a miniature version of the microcontroller that is to be used on UVIS with much of the same programming logic. This would allow for large amounts of preliminary testing without actually having the official robot.

3.2 STAGE 1

There was a significant amount of progress made in the first stage of development. However, it was mainly a learning period to become acquainted with the microcontroller. Afterwards, a number of improvements needed to be made.

3.2.1 Construction

In the first stage of the design process, the first action taken was the construction of a test robot that would simulate the tank style drive train that would be on the UVIS. After brainstorming on possible designs, it was decided that a reverse wheel chair design would best simulate the drive system of the newer robot. With only 2 wheels and motors controlling the motion of the robot, it was decided that the swivel wheels should be placed in the rear for best performance, creating a backwards wheelchair setup. The miniature version of the full microcontroller, servomotors, and a basic erector set made up the Edubot kit that could be used to construct a robot that simulates the functionality of UVIS. The first design of the Edubot consisted of a basic frame made from two angle iron pieces held together by two flat iron pieces, one near the front and one near the back. These were screwed together to form a basic rectangular frame. The next step was to add two drive wheels, one on each side of the chassis. Both of the wheels were placed approximately halfway between the front and back of the robot, and on the outside of the frame thus making the robot move more like a tank rather than having a wheel in each of the four corners that would make the robot drive more like a car. This design does not allow for omni-directional movement like ODIS, but it allows for more accurate tracking because of the simplicity of the movement of a tank-driven system. Once the wheels were positioned in the best spot, a location for the motors had to be determined. Because of the limited range of parts contained in the EduBot kit, it was difficult to set up the robot as was desired. The kit contained no bearings for the wheel axles, so the axles had to be kept as short as possible to allow less drag inside the servomotor. With some trial and error, the axles were lined up with the motor shaft, the mounting plates for the motors, and the frame. Once that had been accomplished, the swivel wheels were attached to the back of the robot for support so the robot would be able to travel with a minimum amount of friction. Having low friction allowed the robot to be more accurate in its movements. Finally, the microcontroller was mounted to the chassis. This was accomplished using standoffs placed just behind the drive wheels so the robot would sit back on the caster wheels and not fall forward. This completed the construction of the test robot, which can be seen

below in Figure 2. After the Edubot model had been built, it was necessary to learn how to program the microcontroller to perform autonomous movement.

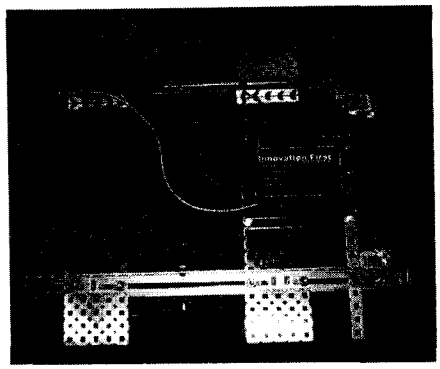


Figure 2: Stage 1 robot.

3.2.2 Programming

The microcontroller on the UVIS and the miniature version were both programmable in C. The EduBot kit supplied a software package with all necessary development tools for writing, testing, and uploading the code. Through reading and research, it was determined that the EduBot's microcontroller was designed for remote control. However, there were still ways to get around this problem using programming manipulations. The power to each motor was controlled in the programming code with integers ranging from zero to 255, with 127 being stopped, 255 being full power forward, and zero being full power backwards. After testing, it was found to have a deadband of approximately 20 over and under the neutral power, meaning that any output power within the range of 97 to 147 did not turn the motors. The microcontroller was set up to perform an endless loop in which every 17ms it checked the input pins, performed user routines, and finally placed the updated output on the PWM (Pulse Width Modulation) pins or the Digital Output Pins. Figure 3 contains an image of the microcontroller with the pin arrangement. Because this control loop ran at a set time, initially the robot was designed to run autonomously based on time. By implementing a counter to increment on each loop, it was possible to get accurately timed movement. From this, the robot was able to go forward and make simple timed turns.

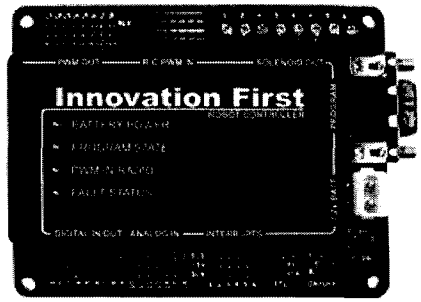


Figure 3: EduBot microcontroller [Inn03].

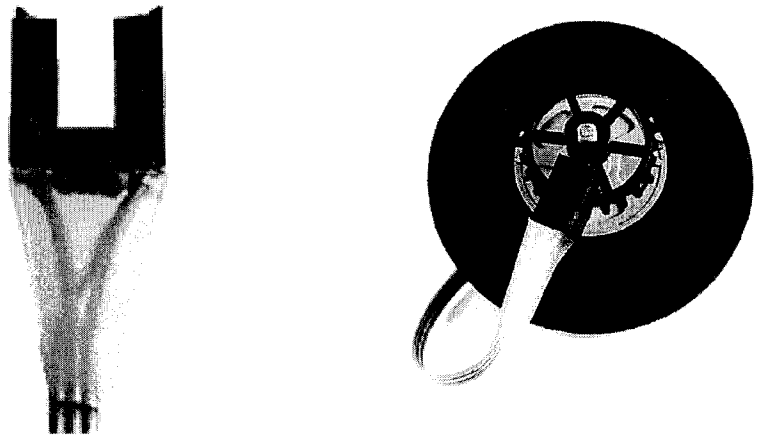
3.2.3 Problems

In this design, a counter was used in the C program to control the total distance traveled by the robot. Because the robot relied on battery power, changes in power supply directly affected the servomotors' rotation speeds. Because the movement was time based, speed changes altered the distance traveled by the robot. This distance measuring issue lead to two problems. First, it meant the difference between scanning a vehicle and seeing a bomb set on the front bumper, or falling short of it and seeing nothing under the vehicle. Second, all turns had to be performed based on exact distances to perform a correct 90-degree turn. If the robot under turned, it could cause it to run into a tire or other object. For these reasons, timed based movement was not accurate enough for the desired applications.

3.3 STAGE 2

3.3.1 Solving Stage 1 Problems

In order to improve upon the design of both the Edubot and the C program, a more accurate way of measuring distance needed to be added. The addition of optical encoders to measure wheel rotations accomplished this. For simplicity, break-beam encoders were chosen. The logic behind the theory of these encoders is simple. When something breaks the infrared beam of light, the output of the encoder becomes zero. Otherwise, it stays high, which is +5 Volts for the EduBot's digital logic system. The break beam encoders used sprockets placed on the axles to determine how many rotations the wheel was making. Whenever a spoke broke the beam, it would increment a counter in the program. Knowing the circumference of the wheel allowed for an easy determination of the distance based on wheel rotations. The robot was then programmed to turn at certain encoder counts. An example of the encoder and how it attaches to a sprocket can be seen below in Figure 4.



(a) (b)

Figure 4: (a) Break-beam encoder, [Car03] (b) implementation of encoder [Car03].

The actual connection of the circuit took some time to perfect. The basic idea of the circuit design came from the datasheet for the break-bream encoder. Below, in Figure 5, it can be seen how to wire a circuit to perform the encoding tasks. The EduBot has a +5 Volt output pin that was used as the Vcc and Input voltages. The connection scheme is as follows:

Pin E: Connect E to K through a 1K ohm load resistor. Then connect to RC Sig on the microcontroller.
Pin C: Connect to +5 Volt RC on the microcontroller.
Pin K: Connect to Blk (Ground) on the microcontroller.
Pin A: Connect Pin A to Pin C through a 100 Ohm resistor.

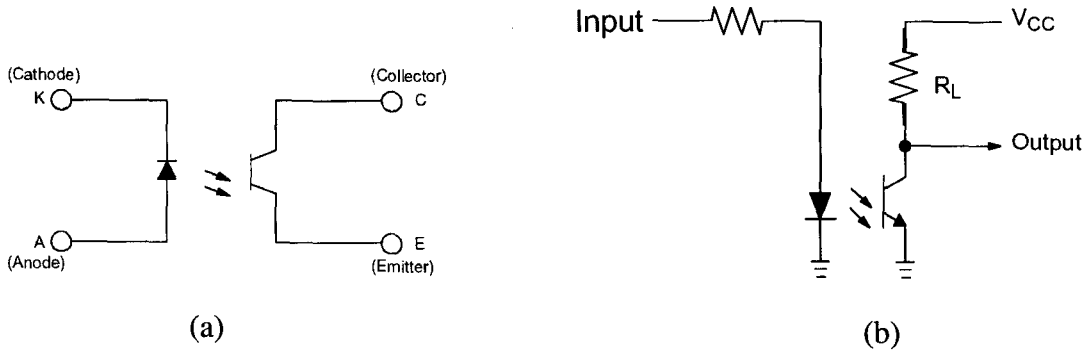


Figure 5: (a) Pin view, [Omr03] (b) circuit view [Omr03].

This setup allowed for an efficient 5 Volt logic encoder. Using optical encoders to measure distance rather than just using a time-based system allowed for much more accurate movement. From this, the test robot was able to move accurate distances as desired. As testing was being performed, it was noticeable that the wheel casters were causing the robot to drag initially, affecting the starting direction. For this reason, free moving ball casters, known as ball transfer systems, were chosen to replace the swivel wheels on the rear to reduce friction and avoid any drag. Figure 6 shows an example of the older casters and the newer improved version.

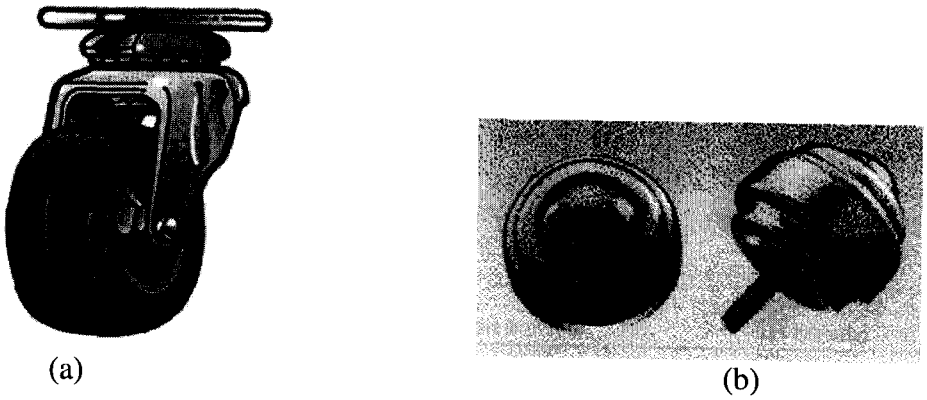


Figure 6: (a) Wheel casters, [Zzo03] (b) stud mount ball transfer systems [Cas03].

The current robot structure did not allow for easy mounting of the encoders. Therefore, the robot was redesigned with the encoders mounted between the motors and wheels and the improved ball casters were added on the rear. The stage 2 robot is pictured below in Figure 7.

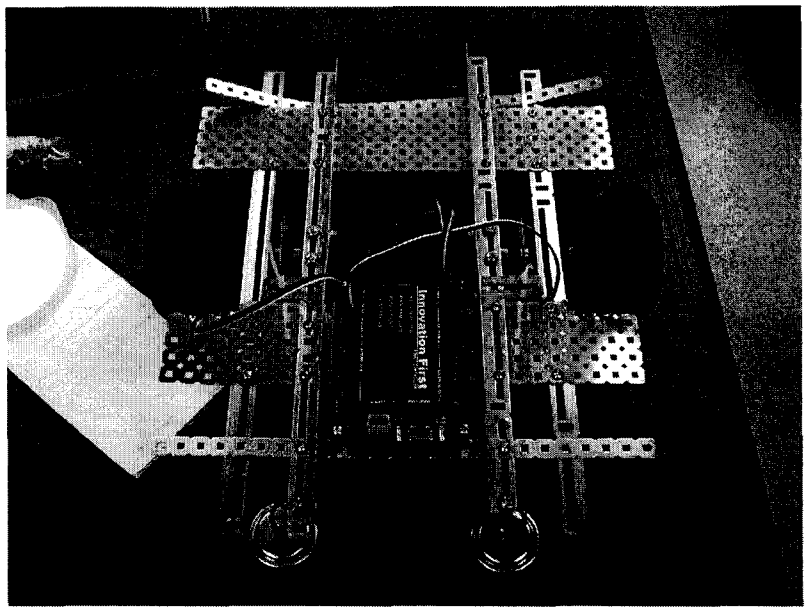


Figure 7: Stage 2 robot.

At this stage of testing, the improved EduBot is capable of accurate distance traveling. Using the encoder to count wheel rotations, the robot can travel around a test box performing multiple 90-degree turns. Despite the high levels of improvement from the encoders and new ball casters, the robot still has significant consistency issues.

3.3.2 Stage 2 Problems

As basic as it may sound, perhaps the most difficult task in automated robotic control is moving consistently in a straight line. Because no two motors are matched, it is nearly impossible to have a robot move in a straight line without any additional aid. Besides this problem, there is also a choice of how the robot should make turns. With the tank driven system being simulated, the EduBot can either perform a 0 degree turn by reversing one of the motors while the other continues forward or a radial turn consisting of stopping the motor on one wheel while the other wheel turns forward until it has reached 90 degrees from its original alignment. For maximum coverage area, the 0 degree turn works well. However, since the motors rarely are matched for rotational speed, it leads to inaccurate turning. Radial turning is independent of motor power which gives it a great advantage over 0 degree turning. Because one wheel simply rotates for an exact number of counts detected by an encoder, the consistency of each turn is relatively high depending on the accuracy of the encoding system. Because the encoders that were formerly on the robot contained only 8

counts per revolution, turns were not been as accurate as desired. Unfortunately, even if the robot were able to make consistent 90-degree radial turns, it still may not have ended up in the correct direction. As can be seen in Figure 8 below, if the EduBot is was not aligned correctly when beginning a 90-degree radial turn, it lead to greater levels of error as the robot continued along the programmed path.

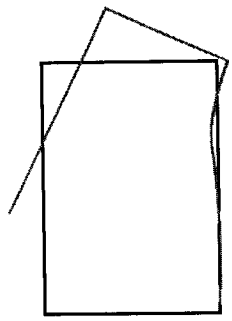


Figure 8: Example of erroneous box travel.

3.4 STAGE 3

3.4.1 Error Reduction

Accuracy and error are directly related. Whenever something needs to be more accurate, the true method for accomplishing that task is reducing error. As the robot was previously designed, the wheels were 4" in diameter and were made of dense foam. As was stated earlier, the sprocket used for encoding contained 8 spokes allowing for 8 counts per wheel revolution using the break-beam encoder.

$$Circumference = \pi \times diameter \tag{1}$$

The circumference of the wheels was approximately 12.6 inches based on Equation 1 above. By knowing the counts per revolution (CPR), the maximum error that could be accumulated between 1 count difference on the left and right encoders could be calculated using Equation 2 below.

$$Error = \frac{Circumference}{CPR} \times 2 \tag{2}$$

By plugging in the previously calculated values, the relative error in movement between counts was around 3.1 inches. The lead to significant levels of drifting before the robot was able to recognize the error. This same error applied to distance measuring. The robot's accuracy in travel was dependent on the error between counts. Reducing error was theoretically easy. One can decrease the circumference, increase the CPR, or do both. In our case, we planned on doing both to provide for a much higher level of accuracy. By switching

to the 2" diameter wheels included in the EduBot kit, the circumference became half of what it was before, increasing accuracy by a factor of 2. The next step was to increase the counts per revolution. Because it was impractical to try to create a 50 or 100 spoke sprocket, 50 CPR optical encoders were purchased from US Digital. With the new values of circumference and CPR plugged into Equation 2, the new error became a mere .25 inches. By implementing the new components on the robot, the accuracy of the robot was increased by more than a factor of 12. However, accuracy in measurement alone did not lead to consistent path following. Without software correction, the robot continued to drift in one direction, due to the mismatched motors. This is where the PID controller came into play.

3.4.2 Proportional Integral Derivative Theory (PID)

"In the PID approach, a controller monitors the error in the system (its deviation from some desired value, or *set point*) and makes corrections based on three criteria. The *Proportional* response is based on the magnitude of the observed error, the *Integral* of that error (error accumulated over time), and the *Derivative* of the error (the rate at which the error changes over time)" [Luc01]. It is easier to understand the concepts by also graphically examining what is happening. First, take a basic example of encoder comparison. For instance, a robot has a left and right encoder that counts wheel rotations. A possible correction could be as follows: if the right counter is greater than the left by a certain amount, decrease the power to the right motor by that amount or a factor of it. This is a good example of how a *Proportional* response works using left-right comparison. The final goal of this correction procedure alone is to obtain equal encoder counts, meaning the wheels have traveled the same distance. However, if the physics is examined more closely, this does not mean the robot will be aligned with its original straight. In fact, it is more likely that the robot will achieve a straight line of travel in a different direction than the original. Figure 9(a) below shows a typical robot path using left-right *Proportional* correction alone. This is the easiest method conceptually to visualize; however, a more common approach for PID controllers is to compare the current speed with a desired speed or *set point* as mentioned earlier. In this case, each wheel acts independently. The microcontroller increases or decreases power depending on the current speed, which is calculated using the optical encoder counts. Figure 9(b) shows an example path of a robot using this type of PID.

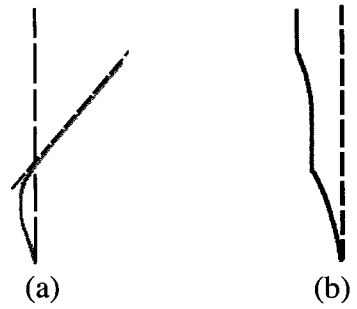


Figure 9: (a) Left-right proportional correction, (b) standard proportional correction.

The next easiest portion of automated software power correction to examine is the *Derivative* component of a PID controller. The *Derivative* is an examination of the rate the error is

changing. For simplicity, in most robotic applications it is simply a difference of current and previous errors. An example of a left-right type of correction would be to analyze the left and right encoder counts at the current time and compare the difference between the left and right counters to the difference at a previous time. For instance, if a robot is veering left by a certain amount, it will record a lower encoder count for the left wheel than the right. If the difference between the encoder counts is growing, it is continuing to veer in that direction. If the power were decreased to the right motor until the difference between the right and left encoder counts equaled the previous difference, straight travel would have been achieved. Unfortunately, like before, the straight line of travel would be in a different direction than the original. The same principles of travel apply to the Derivative portion of the PID using left-right and standard techniques. If the speeds were checked against a desired set value as required in the standard method, the change in errors could be analyzed to obtain the needed information to achieve straight travel by increasing and decreasing power until the difference in errors reached zero. An example of power correction, applicable for both techniques, can be seen in Figure 10 below.

Figure 10: Example of derivative correction.

The last part of power correction is how a robot can find its original desired path. The *Integral* portion of the PID controller keeps up with an accumulation of the error. Equation 3 below shows a simple accumulation of error based on encoder counts using a left-right comparison. Using the more standard method, the accumulation of the error for the left wheel can be found using Equation 4. The CurrentErrorLeft is found by subtracting the current speed of the left wheel from the desired point.

$$Isum = Isum + (Eright - Eleft) \tag{3}$$

$$IerrorLeft = IerrorLeft + CurrentErrorLeft \tag{4}$$

By keeping a running total of the error, the robot is able to tell if it is to the left or right of the original line. When combining all three of the controlling techniques, as the robot makes corrections, its path will become closer to the original path until the measurable error is zero. Figure 11 shows how the robot would correct itself to its original straight reducing the error as it moves.

Figure 11: Full PID correction.

When all three of the methods are combined, an overall output equation is formed. The result of this equation is a power adjustment for a specified motor. The output power is increased by the output amount or decreased by the output amount if the result is negative. The full equation is listed below in Equation 5.

$$Output = Kp * ProportionalError + Kd * (CurrentError - PrevError) + Ki * Ierror \quad (5)$$

The Kp, Kd, and Ki values are tuning constants used to adjust how fast and by what amount the controller corrects power. When all three of these are combined and tuned correctly, the robot will approach a straight path in theory.

3.4.3 Final Status And Problems

Improving upon the Stage 2 design, the current robot has functional 50 CPR optical encoders to accurately measure distance and wheel rotation speed. Several versions of PIDs have been written and implemented with varying levels of success, including hours spent on tuning, using both mathematical and trial and error methods. Despite this fact, the major source of error remains the inconsistent and erratic behavior of the servomotors supplied with the EduBot kit.

One common problem with PIDs is known as integrator windup. This is a condition that results when the integral portion of the controller saturates the power control without actually driving the error towards zero. On robots without true max outputs such as the EduBot, this can lead to the motor switching from forward to backward at high rates. To solve this problem, saturation limits were added to make sure the power of the robot never exceeded the natural min and max, of 0 and 255 respectively, leading to integrator windup. While this successfully resolved the integrator windup issues, the improvement desired by implementing the PID was still voided by the random motor speeds.

The goal in switching to higher count optical encoders and an interrupt driven PID controller was to obtain more accurate movement as well as faster power correction. With the encoding system, it was very easy to check and alter speeds as fast as every 17ms. However, at average speeds for the robot with a 50 CPR encoder, the encoder counts were usually either 1 or 2 for each wheel. If the left wheel had a count of 2 while the right wheel had a count of 1, it was not necessarily moving twice as fast. There was room for error because counts were measured only on rising edges of the encoder output. This inaccurate method lead to errors

nearing 100%. For this reason, the time used to measure speed was increased to 102ms. If a typical count at this sampling rate was 10, the error could at most be 10%, which was acceptable in comparison. However, the problem encountered when trying this method was highly erratic encoder counts. With a constant output power, the encoder counts varied from 7 to 12. This can be seen in Figure 12.

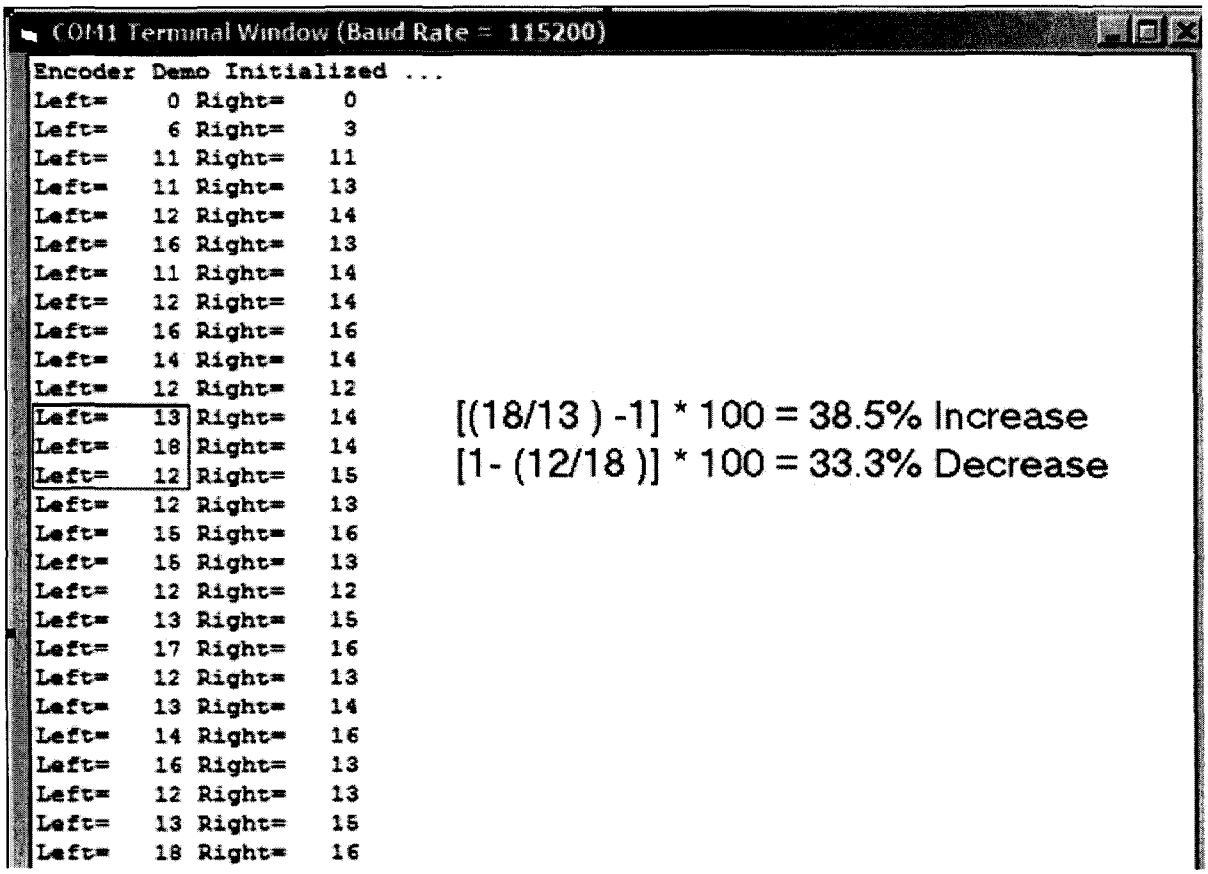


Figure 12: Erratic motor speeds at constant power.

The encoders were checked thoroughly and found to be highly accurate. It was then confirmed that the motors were giving extremely inconsistent output speeds. With variations as high as 5 counts in that short period of time using a constant source of power, the PID was rendered ineffective. To obtain a more reasonable average speed, our sampling period was lowered to 1.02 seconds. In doing this, a variation of 5 counts was obtained on average using a constant power. Even though the count variation remained the same, the actual variation percentage was greatly reduced. From this, an accurate average speed measurement was determined. However, the ability to correct power at high rates was lost. Consequently, power could only be changed as often as the encoder counts, which was every 1.02 seconds. For many applications this could be an acceptable rate. Unfortunately, when traveling short distances with unpredictable motors, it was nearly impossible to travel in a straight line.

While both motors were inconsistent, the left motor had random power dips leading to severe veering towards the left. While a PID could possibly correct this if it could check speeds quickly enough, the 1.02 seconds sampling period that had to be used because of the erratic motor behavior did not allow for the PID to notice the problem until the robot was extremely off course. At that point, the PID couldn't correct to a straight line in time before a turn.

In other attempts to offset the effects of the random motor slow down, weight was added to the right side to relieve stress from the left motor. This was effective to some degree but did not solve the problem. It also lead to initial veering towards the right causing even more problems. In the end, the best test runs obtained used only the break-beam encoders with no PID control.

4 CONCLUSIONS & FUTURE WORK

4.1 CONCLUSIONS

The main goal of this project was to automate the searching process for the new robot to be developed by IRIS Laboratories. In order to add autonomy to the process, it was necessary to program the robot to travel in a defined path. For this to be possible, the robot first had to be able to travel in a straight line consistently. This proved to be the most difficult task of the entire project due mainly to poor quality parts. Many different things were tested to achieve accurate and precise movement by the EduBot. The original optical encoders were upgraded from 8 to 50 CPR for more precise measurement of wheel rotations. In addition, an interrupt driven encoder system was used to measure the high-speed counters, which replaced the timed polling of input pins to check for voltage changes. The design of the robot was also altered many times to reduce friction on the ground as well as the axels. Although, the EduBot kit failed to include many necessary parts for smooth wheel rotation, the robot was able to move with a very low level of drag.

In addition to the hardware changes, the software programming was improved from simple movement to include a PID controller to attempt to account for the types of errors and irregularities of the motors. In theory, the logic behind this method is sound on an engineering basis. Therefore, it should have been able to allow the EduBot to move in a straight line despite mismatched motors. However, even without power correction, the motors varied as much as 38% between checks at a tenth second period. There was a high level of achievement in the project overall. Autonomous movement was accomplished using interrupt driven encoders. A PID was also implemented and partially effective. Despite this fact, there were still many successful autonomous test runs. In the end, it came down to the simple fact that the equipment was too inconsistent to yield completely successful results.

4.2 FUTURE RECOMMENDATIONS

The idea of an automated under-vehicle searching robot remains sound; however, only a quality robot should be used. According to the original project plan, after an initial learning and testing period with the EduBot, that knowledge was going to be applied towards the autonomy of the newer robot dubbed UVIS, which was to be completed by March of 2004. Unfortunately, the robot was never finished in time. If the true fundamentals of engineering were going to be applied, a good recommendation for creating a robot with more consistent motor output would be to use better quality parts. Speed controllers could also help remedy this problem, although this requires significantly more space on a robot for a larger battery. With a top-of-the-line robot that would be produced by IRIS Laboratories, there should no problem implementing the theories and techniques developed throughout the course of this project. The PID logic is reliable and effective. In addition, the processes used to develop autonomous movement worked quite well with the exception of poor motor behavior. Although the EduBot did not perform at the levels hoped, the project was a success overall.

4.3 FUTURE POSSIBILITIES

As time goes on, the need for security will only increase. The future possibilities of a robot like UVIS or ODIS will be abundant. With enough technology and time, multiple types of cameras such as thermal can be added to increase the ability for UVIS to detect threats. Perhaps over time, radiation sensors can even be added to detect nuclear threats. With the development of autonomy, it is foreseeable that a robot of this nature could perform the search completely on its own, without any operator intervention. The future of what the robot may be used for is not yet known, but it is clear that this robot has a purpose in the world of security.

5 BUDGET

Table 1: List of expenses.

Item	Part Number	Price	Quantity	Total Cost
Break-Beam Optical Encoder	EE-SX1061	$1.95	8	$15.60
PWM Extension Cable 12in.	CABLE-PWM-12	$5.00	3	$15.00
Multi- Speed Motor, 7.2V Nominal	EDU-MOTOR-7.2V	$37.50	2	$75.00
Serial Cable DB9 Male to Female	N/A	$2.95	1	$2.95
Optical Kit Encoder 50 CPR	E2-50-250-H-PKG2	$43.00	2	$86.00
Stud Mount Ball Transfer System	SMC 5/16	$4.47	2	$8.94
		Total Cost		$203.49

6 REFERENCES

[Car03] http://www.rec.ri.cmu.edu/education/.

[Cas03] http://www.castersupply.com/NAV/store_ball_transfers.htm.

[ClaOwi03] D. Clark, M. Owings: *Building Robot Drive Trains.* McGraw-Hill, New York, USA, 2003.

[Cso03] http://www.csois.usu.edu/images/products/product_odis1_tn.gif.

[Inn03] http://www.innovationfirst.com.

[Kev03] http://www.kevin.org.

[Luc01] http://www.seattlerobotics.org/encoder/200108/using_a_pid.html.

[Moo03] http://www.engineering.usu.edu/ece/faculty/moorek/denmark-talk.pdf.

[Omr03] http://rocky.digikey.com/WebLib/Omron/Web%20Data/eesx1061.pdf.

[Thi02] http://www.wccusd.k12.ca.us/valleyview/computerlab/misc/ancient-greece-time-line/ancient-greece-time-line.html.

[Zzo03] http://www.zzounds.com/media/feed/large/ERN6101.jpg.